

```
-- broadcast_reg
-- version1 - 1/30/2004
library IEEE;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
library synplify;
use synplify.attributes.all;

entity broadcast_reg is port
  (clock          : in std_logic;
  resetn         : in std_logic;
  state_out      : out std_logic_vector(1 downto 0);
  brcst_in       : in std_logic_vector(5 downto 2);
  brcst_out      : out std_logic_vector(5 downto 2);
  BrcstStr1     : in std_logic);

end broadcast_reg;

architecture behave_rtl of broadcast_reg is
attribute syn_radhardlevel of behave_rtl : architecture is "tmr";
--attribute syn_enum_encoding of behave_rtl : architecture is "sequential";

type state_values is (st0, st1, st2, st3);
signal pres_state : state_values;

begin
  -- fsm register
  state_reg: process (clock, resetn)
    begin
      if (resetn = '0') then
        pres_state      <= st0;
        brcst_out      <= "0000";
      elsif clock'event AND clock = '1' then
        case pres_state is
          when st0 =>
            state_out <= ("00");
            if BrcstStr1 = '1' then
              brcst_out <= brcst_in; -- register data
              pres_state <= st1;
            else
              brcst_out <= "0000";
              pres_state <= st0;
            end if;

          when st1 => -- wait a clock tick to reset
            state_out <= ("01");
            pres_state <= st2;

          when st2 => -- reset output bits
            state_out <= ("10");
            brcst_out <= "0000";
            pres_state <= st3;

          when st3 => --
            state_out <= ("11");
            if BrcstStr1 = '0' then
              pres_state <= st0;
            else -- wait for broadcast strobe to return to '0'
              pres_state <= st3;
            end if;
        end case;
      end if;
    end process;
  
```

C:\Actelprj\top_54sx32aFRONT\hdl\broadcast_reg.vhd

end behave_rtl;